Exercise set 10 – MATH-250 Advanced Numerical Analysis I

The exercise sheet is divided into two sections: quiz and exercises. The quiz will be discussed in the beginning of the lecture on Thursday, May 15. The exercises marked with ($\star$) are graded homework. The exercises marked with **(Python)** are implementation based and can be solved in the Jupyter notebooks which are available on Moodle/Noto. **The deadline for submitting your solutions to the homework is Friday, May 16 at 10h15.**

Quiz

(a) Let $A \in \mathbb{R}^{m\times n}$ with $m \geq n$ and $\mathbf{b} \in \mathbb{R}^m$. If A has rank smaller than n then $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2$ has infinitely many solutions.

☐ True ☐ False

(b) Let $R \in \mathbb{R}^{n\times n}$ be upper triangular then

$$\|R^{-1}\|_2 \leq n \cdot \max\{|r_{11}|^{-1}, \dots, |r_{nn}|^{-1}\}.$$

In particular, this means that $\|R^{-1}\|_2$ can only be large when R has small diagonal entries.

☐ True ☐ False

(c) Consider the problem $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_1$ with $A \in \mathbb{R}^{m\times n}$, $m \geq n$, and $\mathbf{b} \in \mathbb{R}^m$. Then there is always a minimizer $\mathbf{x}^\star$ such that the residual $A\mathbf{x}^\star - \mathbf{b}$ has at least one zero entry.

☐ True ☐ False

Exercises

Problem 1. (Python)

(a) Write a function `gradient` in Python that solves $A\mathbf{x} = \mathbf{b}$ using the Gradient method. Your function should take as inputs

- The symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$
- The right hand side $\mathbf{b} \in \mathbb{R}^n$
- The tolerance *rtol*

The function should output

- A vector $\hat{\mathbf{x}} \in \mathbb{R}^n$ such that $\frac{\|A\hat{\mathbf{x}} - \mathbf{b}\|_2}{\|\mathbf{b}\|_2} < rtol$.
- The number of iterations required to achieve a relative error smaller than *tol*.
- A vector consisting of the norms of the residuals at each iteration $\|\mathbf{r}^{(k)}\|_2$.

(b) Apply your function to the two linear systems $A\mathbf{x} = \mathbf{b}$ where

(1)

$$A_1 = \begin{pmatrix} 3 & 2 \\ 2 & 6 \end{pmatrix}, \quad \mathbf{b}_1 = \begin{pmatrix} 2 \\ -8 \end{pmatrix}$$

(2) $A_2 \in \mathbb{R}^{1024 \times 1024}$ with right hand side $\mathbf{b}_2 \in \mathbb{R}^{1024}$ generated by the following code

```
import numpy as np
import scipy as sp

n = 32
a = sp.sparse.diags([-1, 2, -1], [-1, 0, 1], shape=(n, n))
I = sp.sparse.eye(n)
A_2 = sp.sparse.kron(I, a) + sp.sparse.kron(a, I)

def f(x, y):
    return -(12 * x ** 2 - 6 * x) * y * (y - 1) - 2 * x ** 3 * (x - 1)

t = np.tile(np.arange(1, n + 1) / (n + 1), (n, 1))
x = t.T.flatten()
y = t.flatten()
b_2 = f(x, y) / ((n + 1) ** 2)
```

For $\mathsf{tol} = 10^{-8}$ plot the norm of the residuals $\|\mathbf{r}^{(k)}\|_2$ versus k. Compare your method with the Conjugate Gradient method. You may use the built-in conjugate gradient method `scipy.sparse.linalg.cg`/`sp.sparse.linalg.cg` in Python.[1]

Problem 2.

Prove Lemma 6.2 in the lecture notes. That is, show that if $A \in \mathbb{R}^{m \times n}$ has rank n then $A^\top A$ is symmetric positive definite.

Problem 3. Assume that you are given data $t_1, \ldots, t_m \in \mathbb{R}$ and $y_1, \ldots, y_m \in \mathbb{R}$. Suppose that x_1 and x_2 are chosen such that

$$\sum_{i=1}^{m} (x_1 + x_2 t_i - y_i)^2$$

is minimized. Further, define $\hat{y}_i = x_1 + x_2 t_i$ and $r_i = y_i - \hat{y}_i$ for $i = 1, \cdots, m$. Show that

(a) $\sum\limits_{i=1}^{m} r_i = 0$

(b) Let $\bar{t} = \frac{1}{m} \sum\limits_{i=1}^{m} t_i$ and $\bar{y} = \frac{1}{m} \sum\limits_{i=1}^{m} y_i$. Then $\bar{y} = x_1 + x_2 \bar{t}$.

(c) $\sum\limits_{i=1}^{m} t_i r_i = 0$

(d) $\sum\limits_{i=1}^{m} \hat{y}_i r_i = 0$

[1]Note that the function `scipy.sparse.linalg.cg` in Python does not return the norm of the residual for each iteration. An ugly solution is to call the function multiple times in a loop, increasing the maximum number of iterations until convergence, and computing by hand the residual norm each time.

($\star$) **Problem 4.**

Let $A, P \in \mathbb{R}^{n\times n}$ be symmetric and positive definite matrices. Consider the linear system

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

(a) We denote the Cholesky factorisation $P = LL^T$, where L is a lower triangular matrix.

Derive a relation between the solution x of (1) and the solution $\tilde{x}$ to

$$\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}, \tag{2}$$

where $\tilde{A} = L^{-1}AL^{-T}$ and $\tilde{\mathbf{b}} = L^{-1}\mathbf{b}$.

(b) Apply the gradient method to the linear system (2) and show that it is equivalent to an iterative method given by the update

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k P^{-1}(\mathbf{b} - A\mathbf{x}^{(k)}). \tag{3}$$

Starting from the expression for α_k given in the lecture notes, derive an expression for α_k that only involves A and P^{-1}. In particular, it should not involve $\tilde{A}$ or the Cholesky factor L.

(c) Define $\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}$ the residual after the k-th iteration.

Show that

$$\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k+1)} \rangle_{P^{-1}} = 0, \quad k \geq 1$$

where $\langle \mathbf{y}, \mathbf{z} \rangle_{P^{-1}} = \mathbf{y}^\top P^{-1}\mathbf{z}$.

(d) The method (3) is called the preconditioned gradient method.

Write a Python function `gradient(A, b, P)` that implements the preconditioned gradient method for a matrix A, a vector $\mathbf{b}$, and a preconditioning matrix P. Stop the iteration once the relative error $\frac{\|\mathbf{r}^{(k)}\|_2}{\|\mathbf{b}\|_2}$ is smaller than 10^{-6}, and return the solution $\mathbf{x}^{(k)}$, the residual norms $\|\mathbf{r}^{(1)}\|_2, \|\mathbf{r}^{(2)}\|_2, \dots, \|\mathbf{r}^{(k)}\|_2$ and the number of iterations k executed to reach the solution. Ensure that if no preconditioner is given in the function arguments then the unpreconditioned gradient method is run.

(e) Run the gradient method for the system given by

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 0 \\ -1 \\ 2 \end{pmatrix}$$

without any preconditioning. Clearly print the number of iterations.

(f) On Moodle we provide a matrix $A \in \mathbb{R}^{n\times n}$ in the file `matrix10.npz`. Load this matrix using SciPy `sparse`'s `load_npz` (or `scipy.sparse.load_npz` if you are not using our provided Jupyter notebook), and define the right-hand side $\mathbf{b} = [1, 1, \dots, 1]^\top$ of appropriate size.

Run the preconditioned gradient method with the preconditioners

- $P_1 = I_{n\times n}$,

- $P_2 = \mathrm{diag}(a_{11}, a_{22}, \ldots, a_{nn})$, and
- $P_3 = LU$.

Plot the residual norms $\|\mathbf{r}^{(i)}\|_2$ for increasing numbers of iterations for the preconditioners P_1, P_2, and P_3. Use a single plot for all three preconditioners.

Hint: Use `sps.linalg.spilu` to compute the incomplete LU factorisation of the sparse matrix A (use `scipy.sparse.linalg.spilu` if you do not want to use the notebooks on Moodle). This method returns a `SuperLU` object, meaning you can directly call its member function `solve` on a matrix M to compute $P^{-1}M$.